Classification of CIFAR-10 Dataset

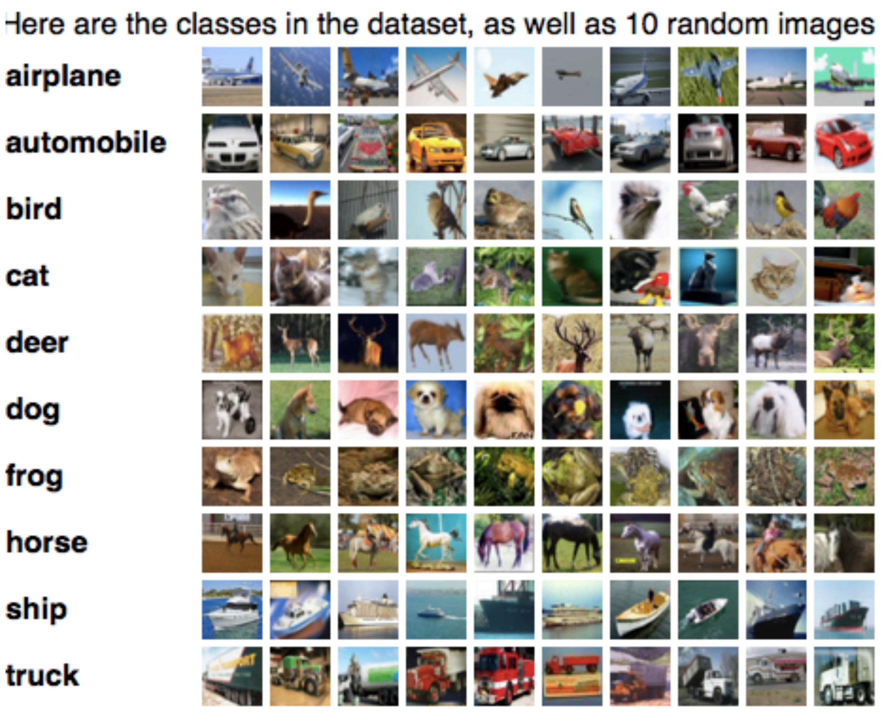
Introduction

Deep Learning has become very popular in areas like image recognition. We are gradually switching all the tasks that were once handled only by humans, to machines, as machines can process a task very quickly, reducing cost and man-power.
As part of this assignment, I have tried to classify the CIFAR-10 dataset which consists of 10 different classes using MLP and CNN.

Dataset

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

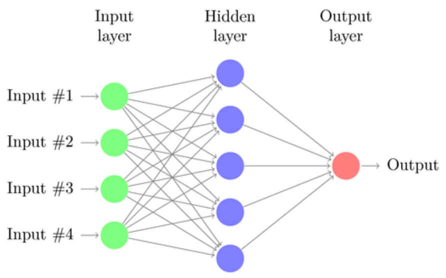
The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.



Here are the classes in the dataset, as well as 10 random images from each:

Algorithms

Multi-Layer Perceptron - A multilayer perceptron (MLP) is a class of feed-forward artificial neural network. An MLP consists of, at least, three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called back-propagation for training.

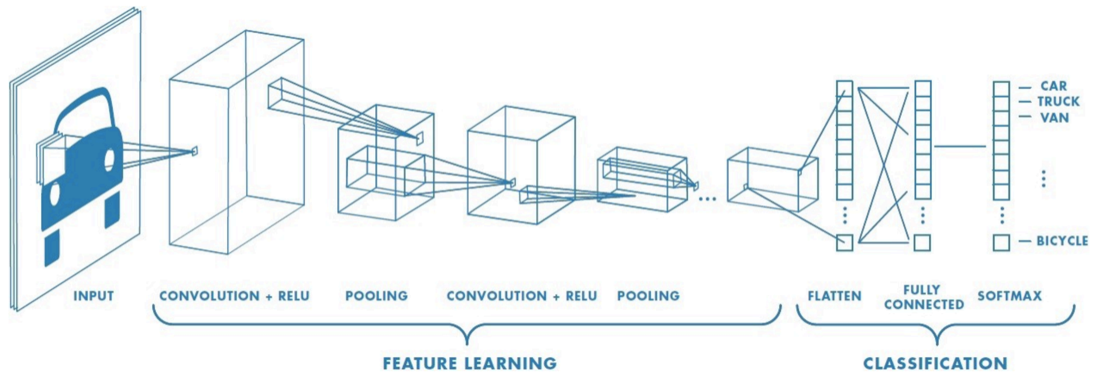


Accuracy table for 50,000 training images, and 10,000 test images.

Algorithms	Parameters	Accuracy
Multi Layer Perceptron	**Case 1: Base Model** 1. No of epochs = 15 2. Batch size = 100 3. Number of neurons in a layer = 256 4. Number of layers = 3 5. Learning rate = 0.01 6. Activation functions = ReLu 7. Dropout rates = NA	50.07%
	Case 2: Increasing Number of Epochs 1. No of epochs = 20 2. Batch size = 100 3. Number of neurons in a layer = 256 4. Number of layers = 3 5. Learning rate = 0.01 6. Activation functions = ReLu 7. Dropout rates = NA	49%
	Case 3: Increasing the batch size 1. No of epochs = 10 2. Batch size = 160 3. Number of neurons in a layer = 256 4. Number of layers = 3 5. Learning rate = 0.01 6. Activation functions = ReLu 7. Dropout rates = NA	51.26%

	Parameters	
Multi Layer Perceptron	**Case 5: Increasing the number of Layers and neurons** 1. No of epochs = 10 2. Batch size = 100 3. Number of neurons in a layer = 350 4. Number of layers = 4 5. Learning rate = 0.01 6. Activation functions = ReLu 7. Dropout rates = NA	50.13%
	Case 6: Increasing Learning Rate 1. No of epochs = 10 2. Batch size = 100 3. Number of neurons in a layer = 256 4. Number of layers = 3 5. Learning rate = 0.05 6. Activation functions = ReLu 7. Dropout rates = NA	42%
	Case 7: Increasing the Activation Function 1. No of epochs = 10 2. Batch size = 100 3. Number of neurons in a layer = 256 4. Number of layers = 3 5. Learning rate = 0.01 6. Activation functions = sigmoid 7. Dropout rates = NA	47.64%
	Case 8: Adding a dropout Dropout rates = 0.2	48.65%

Convolutional Neural Network - In machine learning, a CNN is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery. CNNs use relatively little pre-processing compared to other image classification algorithms.



Accuracy table for 50,000 training images, and 10,000 test images.

Algorithms	Parameters	Accuracy
Convolutional Neural Network	Base Model – Without Image Augmentation	69.19%
	Case 1 : Rotation by 60 Degrees and flipping the images Horizontally and vertically	64%
	Case 2: Setting Up Feature - wise standard normalization as true	45.94%
	Case 3: Rotation by 10 Degrees and height shift of 0.2	77.98%

Conclusion

After fine-tuning/changing some parameters of MLP and CNN, I can say CNN gave the best prediction, i.e, 77.98%, after applying image augmentation of image rotation by 10 degrees and height shift of 0.02. However, accuracy might change depending on how we tune our parameters. More parameters need to be studied to further improve the accuracy and more data should be used to train the model.

References

https://github.com/fchollet/keras/blob/master/examples/cifar10_cnn.py

https://keras.io/getting-started/faq/

https://github.com/fchollet/keras/blob/master/examples/mnist_mlp.py